(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号
特開2001－188688
（P2001－188688A）
(43)公開日　平成13年７月10日(2001.7.10)

(51)Int.Cl.[7]		識別記号	FI		テーマコード(参考)
Ｇ０６Ｆ	11/00	３５０	Ｃ０６Ｆ 11/00	３５０Ｅ	５Ｂ０４２
	11/30	３０５	11/30	３０５Ｊ	

審査請求　未請求　請求項の数２　ＯＬ　（全 ６ 頁）

(21)出願番号　特願2000－245(P2000－245)

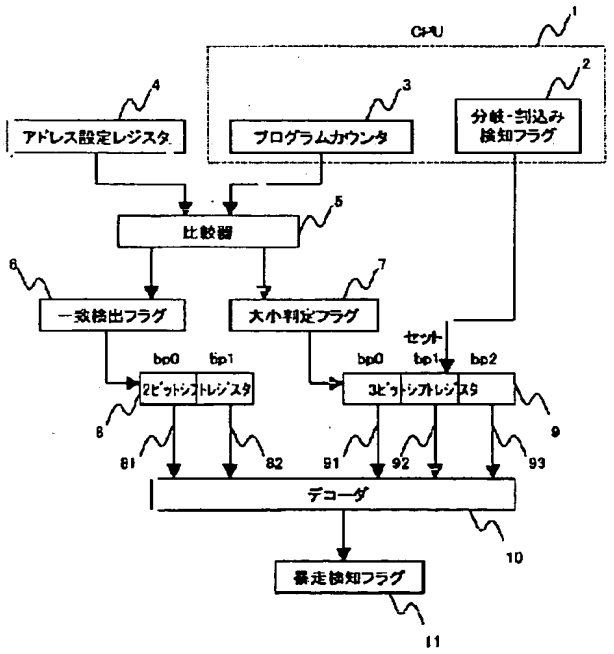
(22)出願日　平成12年１月５日(2000.1.5)

(71)出願人　000005821
松下電器産業株式会社
大阪府門真市大字門真1006番地
(72)発明者　山本　有紀
大阪府門真市大字門真1006番地　松下電器産業株式会社内
(72)発明者　多那瀬　寛
大阪府門真市大字門真1006番地　松下電器産業株式会社内
(74)代理人　100097445
弁理士　岩橋　文雄　（外２名）

Ｆターム(参考)　5B042 GA13 JJ13 LA10

(54)【発明の名称】　暴走検知回路

(57)【要約】
【課題】　ノイズや電圧の急激な変動などにより、プログラムカウンタの値がデータ化けを起こすと、命令コードのフェッチミスが発生し、ＣＰＵが誤った命令コードを処理しながらプログラムが進んでしまう。本発明は、誤った命令コードが処理されたことを検知することを目的とする。
【解決手段】　可変長命令の命令境界となるアドレスを予めアドレス設定レジスタ４に設定しておく。前記設定値とプログラムカウンタ３とを比較器５により常に比較し、プログラムカウンタ３の値が正しい順序通りに変化していることを確認し、プログラムカウンタ３が異常値をとったときには、これを検出することができる。本発明では、分岐・割込み命令を含むプログラムにおいても暴走検知が有効である。



【特許請求の範囲】
【請求項１】予めデータを格納するための第１の記憶装置と、前記第１の記憶装置の値とプログラムカウンタの値を比較して大小判定および一致検出を行う比較器と、前記大小判定および一致検出結果を格納するための第２、第３の記憶装置と、前記第２、第３の記憶装置の値をデコードするデコーダからなる、マイクロコンピュータの暴走検知回路。
【請求項２】前記予めデータを格納するための第1の記憶装置を複数個設ける手段を備えた、マイクロコンピュータの暴走検知回路。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、マイクロコンピュータの誤動作を検知するための機構に関するものである。
【０００２】
【従来の技術】一般に、マイクロコンピュータは、ＣＰＵがメモリに格納された命令コードを順序通り実行することで処理が進んでいく。しかし、ノイズや電圧の急激な変動の影響を受けることでプログラムカウンタの異常や命令コードのフェッチミスが発生した場合には、ＣＰＵは正しく命令を実行できなくなる。従来の技術では、ウオッチドックタイマによる一定間隔でのＣＰＵの監視や未定義命令の検知により、ＣＰＵの誤動作を極力防止してきた。
【０００３】
【発明が解決しようとする課題】可変長命令アーキテクチャを持つマイクロプロセッサがノイズを受けて暴走するメカニズムの一つに、プログラムカウンタの値が異常になる動作がある。プログラムカウンタに異常が起こると、図９に示すように誤った命令境界からデータをとり込んでしまう。ところが、従来のウオッチドックタイマによる一定間隔での暴走検出機構であると、暴走が検出されるまでに時間がかかり、その間ＣＰＵは誤動作してしまう。また、未定義命令の検知による暴走検出機構であると、暴走後にフェッチした命令コードが前後の命令コードとの関係に違反していなければこれを検知することができず、ＣＰＵは誤った命令を処理しながら動作してしまう。
【０００４】本発明は上記課題を解決するために、誤った命令コードが処理されたことを検知することを目的とする。
【０００５】
【課題を解決するための手段】上記課題を解決するため、本発明では、図１０に示すように可変長命令の命令境界となるアドレスを予め設定しておき、プログラムカウンタと常に比較を行う。従って、本発明の実行形態には、前記アドレスを設定するレジスタと、前記アドレスとプログラムカウンタとを比較する比較器とを備える。また、プログラムカウンタが前記アドレスに対して異常値となった場合、これを検知するためのシフトレジスタとデコーダとを備える。
【０００６】
【発明の実施の形態】（実施の形態１）以下に本発明の第１の実施の形態について、図１を用いて説明する。図１は本発明の第１の実施の形態における暴走検知回路の構成を示すブロック図である。
【０００７】１はマイクロコンピュータのＣＰＵ、２は分岐命令または割込み命令が発生したときにＣＰＵ１内で検知される分岐・割込み検知フラグである。３はＣＰＵ１内のプログラムカウンタであり、４は予め命令境界アドレスを設定しておくためのアドレス設定レジスタである。比較器５はプログラムカウンタ３とアドレス設定レジスタ４の値を比較する。
【０００８】一致検出フラグ６には、比較器５の比較結果から、プログラムカウンタ３の値とアドレス設定レジスタ４の値が一致したとき‘１’、それ以外は‘０’が設定される。同様に、大小判定フラグ７には、プログラムカウンタ３の値がアドレス設定レジスタ４の値と一致したときか、大きいとき‘１’、それ以外は‘０’が設定される。
【０００９】８は一致検出フラグ６をラッチする２ビットシフトレジスタであり、９は大小判定フラグをラッチする３ビットシフトレジスタである。３ビットシフトレジスタ９の２ビット目に相当するｂｐ１は、分岐・割込み検知フラグ２によって、ＣＰＵが分岐処理や割込み処理を実行する際にセットされる。２ビットシフトレジスタ８の出力信号８１、８２及び３ビットシフトレジスタ９の出力信号９１、９２、９３はデコーダ１０によってデコードされる。デコーダ１０は出力信号９１、９２、９３、８１、８２が‘１０００’となったとき暴走検知フラグ１１を発生する。
【００１０】以下に、図２～図４を参照して、暴走検知のフローを説明する。図２～図４はアドレス設定レジスタ４に命令境界アドレス‘５３Ｈ’（１６進数表記）を設定した例であり、ＣＰＵの動作状態が図２は分岐の成立や割込みの発生のない状態での正常動作の場合を、図３は異常動作の場合を、図４は分岐命令が成立した場合を示している。
【００１１】（１）分岐の成立や割込みの発生のない状態での正常動作の場合
ＣＰＵが正常に動作している場合について、図２を参照して説明する。このとき、プログラムカウンタ３の値はアドレス設定レジスタ４の設定値に対して、設定値より小さい値、設定値と同じ値、設定値より大きい値の順に変化する。
【００１２】プログラムカウンタ３がアドレス設定レジスタ４の設定値よりも小さい値のときには、大小判定フラグ７および一致検出フラグ６が‘０’であり、３ビッ

トシフトレジスタ９、２ビットシフトレジスタ８ともに全ビット‘０’である。プログラムカウンタ３の値がアドレス設定レジスタ４の設定値‘５３Ｈ’と一致したとき、大小判定フラグ７及び一致検出フラグ６が‘１’となり３ビットシフトレジスタ９、２ビットシフトレジスタ８のｂｐ０に‘１’がラッチされる。プログラムカウンタ３の値がアドレス設定レジスタ４の設定値よりも大きい値になると、大小判定フラグ７は‘１’、一致検出フラグ６は‘０’となり、図２に示すように、３ビットシフトレジスタ９は‘１１０’、２ビットシフトレジスタ８は‘０１’となる。

【００１３】従ってこのフローでは、出力信号９１、９２、９３、８１、８２が‘１０００’となることはなく、暴走検知フラグ１１はアクティブにはならない。

【００１４】（２）異常動作の場合

ＣＰＵに異常動作が発生した場合について、図３を参照して説明する。プログラムカウンタ３がノイズなどにより異常値をとった場合、プログラムカウンタ３の値は、アドレス設定レジスタ４の設定値に一致せず通過してしまう。

【００１５】プログラムカウンタ３がアドレス設定レジスタ４の設定値よりも小さい値のときには、大小判定フラグ７及び一致検出フラグ６が‘０’であり、３ビットシフトレジスタ９、２ビットシフトレジスタ８ともに全ビット‘０’である。プログラムカウンタ３の値がアドレス設定レジスタ４の設定値に一致することなく大きい値になると、大小判定フラグ７は‘１’、一致検出フラグ６は‘０’となり、図３に示すように、３ビットシフトレジスタ９は‘１００’、２ビットシフトレジスタ８は‘００’となる。

【００１６】従ってこのフローでは、プログラムカウンタ３の値がアドレス設定レジスタ４の設定値を通過した直後に、出力信号９１、９２、９３、８１、８２が‘１００００’となり、暴走検知フラグ１１がアクティブになることから、異常命令処理による暴走を検知できる。

【００１７】（３）分岐命令が成立した場合

ＣＰＵの動作状態に分岐が発生した場合について、図４を参照して説明する。このとき、プログラムカウンタ３の値は、アドレス設定レジスタ４の設定値に一致することなく通過してしまう場合がある。

【００１８】プログラムカウンタ３がアドレス設定レジスタ４の設定値よりも小さい値のときには、大小判定フラグ７及び一致検出フラグ６が‘０’であり、３ビットシフトレジスタ９、２ビットシフトレジスタ８ともに全ビット‘０’である。分岐が発生すると、分岐・割込み検知フラグ２により３ビットシフトレジスタ９のｂｐ１が‘１’にセットされ、プログラムカウンタ３の値がアドレス設定レジスタ４の設定値に一致することなく、大きい値になると、大小判定フラグ７は‘１’、一致検出フラグ６は‘０’となり、図４に示すように、３ビットシフトレジスタ９は‘１０１’、２ビットシフトレジスタ８は‘００’となる。

【００１９】従ってこのフローでは、出力信号９１、９２、９３、８１、８２が‘１０００’となることはなく、暴走検知フラグ１１はアクティブにはならない。割込みが発生した場合でも、同様の処理が行われることは自明である。

【００２０】以上（１）、（２）、（３）より、本実施形態では、プログラムカウンタ３が異常値をとると、暴走検知フラグ１１がアクティブになることから、暴走検知が可能となる。

【００２１】（実施の形態２）以下に本発明の第２の実施の形態について、図５を用いて説明する。第２の実施の形態は、複数の命令境界アドレスを設定することのできる機構である。図５は本発明の第２の実施の形態における暴走検知回路の構成を示すブロック図である。

【００２２】１はマイクロコンピュータのＣＰＵであり、２は分岐命令または割込み命令が発生したときにＣＰＵ１内で検知される分岐・割込み検知フラグである。３はＣＰＵ１内のプログラムカウンタであり、４１～４４は予め命令境界アドレスを設定しておくためのアドレス設定レジスタである。アドレス設定レジスタは４個設け、プログラムカウンタ３の最上位２ビットで区別されるメモリ領域ごとに１つの命令境界アドレスを設定するものとする。

【００２３】アドレス設定レジスタ４１には最上位２ビットが‘００’の命令境界アドレスを、アドレス設定レジスタ４２には最上位２ビットが‘０１’の命令境界アドレスを、アドレス設定レジスタ４３には最上位２ビットが‘１０’の命令境界アドレスを、アドレス設定レジスタ４４には最上位２ビットが‘１１’の命令境界アドレスを設定する。

【００２４】セレクタ１２はプログラムカウンタ３の最上位２ビットが‘００’のときアドレス設定レジスタ４１を、‘０１’のときアドレス設定レジスタ４２を、‘１０’のときアドレス設定レジスタ４３を‘１１’のときアドレス設定レジスタ４４を選択することで、プログラムカウンタ３と同じ領域の命令境界アドレス設定値を選択することができる。セレクト変更フラグ１３には、セレクタ１２でのアドレス設定レジスタ４１～４４の選択が変化したときに‘１’、それ以外は‘０’が設定される。

【００２５】比較器５はプログラムカウンタ３とセレクタ１２の値を比較する。一致検出フラグ６には、比較器５の出力結果から、プログラムカウンタ３の値とセレクタ１２の値が一致したとき‘１’、それ以外は‘０’が設定される。同様に、大小判定フラグ７には、プログラムカウンタ３の値がセレクタ１２の値と一致したときか、大きいとき ‘１’、それ以外は‘０’が設定される。８は一致検出フラグ６をラッチする２ビットシフト

レジスタであり、9は大小判定フラグをラッチする3ビットシフトレジスタである。3ビットシフトレジスタ9のｂｐ1は分岐・割込み検知フラグ2によって、ＣＰＵが分岐処理や割込み処理を実行する際にセットされ、ｂｐ0及びｂｐ2はセレクト変更フラグ13によってクリアされる。2ビットシフトレジスタ8の出力信号81、82及び3ビットシフトレジスタ9の出力信号91、92、93はデコーダ10によってデコードされる。デコーダ10は出力信号91、92、93、81、82が'10000'となったとき暴走検知フラグ11を発生する。

【0026】命令境界アドレスは、プログラムカウンタ3の上位ｎビットで区別されるメモリ領域ごとに設定できるので、アドレス設定レジスタは2ｎ個設けることも可能である。

【0027】以下に、図6～図8を参照して、暴走検知のフローを説明する。図6～図8は命令境界アドレスとして、アドレス設定レジスタ41に'072H'、アドレス設定レジスタ42に'114H'、アドレス設定レジスタ43に'243H'、アドレス設定レジスタ44に'330H'を設定した例であり、ＣＰＵの動作状態が図6は分岐の成立や割込みの発生のない状態での正常動作の場合を、図7は異常動作の場合を、図8は分岐命令が成立した場合を示している。

【0028】(1)分岐の成立や割込みの発生のない状態での正常動作の場合

ＣＰＵが正常に動作している場合について、図6を参照して説明する。このとき、プログラムカウンタ3の値は、セレクタ12の選択値に対して、選択値より小さい値、選択値、選択値より大きい値の順に変化する。

【0029】図6ではプログラムカウンタ3が'050H'から始まる例を示している。このとき、プログラムカウンタ3の最上位2ビット'00'により、セレクタ12ではアドレス設定レジスタ41が選択されているので、セレクタ12の選択値は'072H'である。プログラムカウンタ3がセレクタ12の選択値よりも小さい値のときには、大小判定フラグ7および一致検出フラグ6が'0'であり、3ビットシフトレジスタ9、2ビットシフトレジスタ8ともに全ビット'0'である。プログラムカウンタ3の値がセレクタ12の選択値'072H'と一致したとき、大小判定フラグ7及び一致検出フラグ6が'1'となり3ビットシフトレジスタ9、2ビットシフトレジスタ8のｂｐ0に'1'がラッチされる。

【0030】プログラムカウンタ3の値がセレクタ12の選択値よりも大きい値になると、大小判定フラグ7は'1'、一致検出フラグ6は'0'となり、図6に示すように、3ビットシフトレジスタ9は'110'、2ビットシフトレジスタ8は'01'となる。

【0031】さらにプログラムが進んで、プログラムカウンタ3の最上位2ビットが'00'から'01'に変化するとセレクト変更フラグ13が'1'にセットされることで、3ビットシフトレジスタ9のｂｐ0及びｂｐ2が'0'にクリアされ、セレクタ12の選択がアドレス設定レジスタ42に変化する。

【0032】従ってこのフローでは、出力信号91、92、93、81、82が'10000'となることはなく、暴走検知フラグ11はアクティブにはならない。

【0033】(2)異常動作の場合

ＣＰＵに異常動作が発生した場合について、図7を参照して説明する。プログラムカウンタ3がノイズなどにより異常値をとった場合、プログラムカウンタ3の値は、セレクタ12の選択値に一致せず通過してしまう。

【0034】図6の正常動作の場合と同様に、プログラムカウンタ3が'110H'まで進んだとする。この後、プログラムカウンタ3が異常値をとり、セレクタ12の選択値'114H'に一致することなく大きい値になると、大小判定フラグ7は'1'、一致検出フラグ6は'0'となり、図7に示すように、3ビットシフトレジスタ9は'100'、2ビットシフトレジスタ8は'00'となる。

【0035】従ってこのフローでは、プログラムカウンタ3の値がセレクタ12の選択値を通過した直後に、出力信号91、92、93、81、82が'10000'となり、暴走検知フラグ11がアクティブになり、異常命令処理による暴走を検知できる。

【0036】(3)分岐命令が成立した場合

ＣＰＵの動作状態に分岐が発生した場合について、図8を参照して説明する。このとき、プログラムカウンタ3の値は、セレクタ12の選択値に一致することなく通過してしまう場合がある。

【0037】図6の正常動作の場合と同様に、プログラムカウンタ3が'081H'まで進んだとする。この後、分岐が発生すると、プログラムカウンタ3の最上位2ビットが'00'から'10'に変化するとセレクト変更フラグ13が'1'にセットされることで、3ビットシフトレジスタ9のｂｐ0及びｂｐ2が'0'にクリアされ、さらに、分岐・割込み検知フラグ2により、ｂｐ1が'1'にセットされる。また、セレクタ12の選択がアドレス設定レジスタ43に変化する。この分岐により、プログラムカウンタ3の値がセレクタ12の選択値に一致することなく、大きい値になると、大小判定フラグ7は'1'、一致検出フラグ6は'0'となり、図8に示すように、3ビットシフトレジスタ9は'101'、2ビットシフトレジスタ8は'00'となる。

【0038】従ってこのフローでは、出力信号91、92、93、81、82が'10000'となることはなく、暴走検知フラグ11はアクティブにはならない。割込みが発生した場合でも、同様の処理が行われることは自明である。

【００３９】以上（１）、（２）、（３）より、本実施形態では、アドレス設定レジスタを複数個設けることで、精度の良い暴走検知が可能となる。

【００４０】

【発明の効果】以上の実施の形態から分かるように本発明によれば、ノイズや、急激な電圧変動などにより、プログラムカウンタが異常値をとったとき、暴走検知を行うことができるので、ＣＰＵの誤動作を防止することができる。

【図面の簡単な説明】

【図１】本発明の第１の実施の形態における暴走検知回路の構成を示すブロック図

【図２】本発明の第１の実施の形態における暴走検知回路の動作フローの一例（分岐の成立や割込みの発生のない状態での正常動作の場合）を示す図

【図３】本発明の第１の実施の形態における暴走検知回路の動作フローの一例（異常動作の場合）を示す図

【図４】本発明の第１の実施の形態における暴走検知回路の動作フローの一例（分岐命令が成立した場合）を示す図

【図５】本発明の第２の実施の形態における暴走検知回路の構成を示すブロック図

【図６】本発明の第２の実施の形態における暴走検知回路の動作フローの一例（分岐の成立や割込みの発生のない状態での正常動作の場合）を示す図

【図７】本発明の第２の実施の形態における暴走検知回路の動作フローの一例（異常動作の場合）を示す図

【図８】本発明の第２の実施の形態における暴走検知回路の動作フローの一例（分岐命令が成立した場合）を示す図

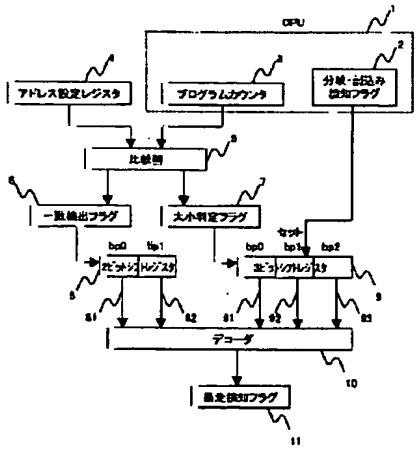
【図９】プログラムカウンタ異常による異常命令処理の例を示す図

【図１０】本発明のプログラムカウンタ異常による暴走検知の概念図

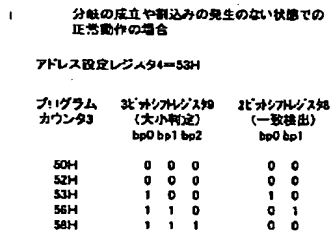
【符号の説明】

１　ＣＰＵ
２　分岐・割込み検知フラグ
３　プログラムカウンタ
４　アドレス設定レジスタ
５　比較器
６　一致検出フラグ
７　大小判定フラグ
８　２ビットシフトレジスタ
９　３ビットシフトレジスタ
１０　デコーダ
１１　暴走検知フラグ
１２　セレクタ
１３　セレクト変更フラグ
４１　アドレス設定レジスタ
４２　アドレス設定レジスタ
４３　アドレス設定レジスタ
４４　アドレス設定レジスタ
８１　２ビットシフトレジスタ８出力信号
８２　２ビットシフトレジスタ８出力信号
９１　３ビットシフトレジスタ９出力信号
９２　３ビットシフトレジスタ９出力信号
９３　３ビットシフトレジスタ９出力信号

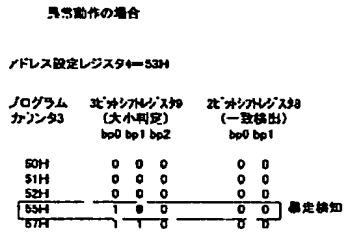
【図１】



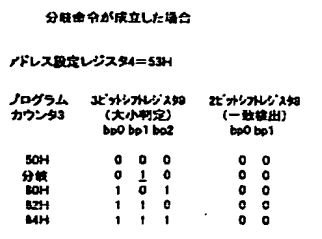
【図２】



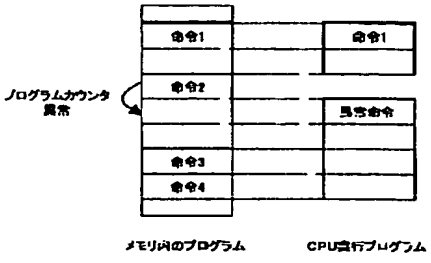
【図３】



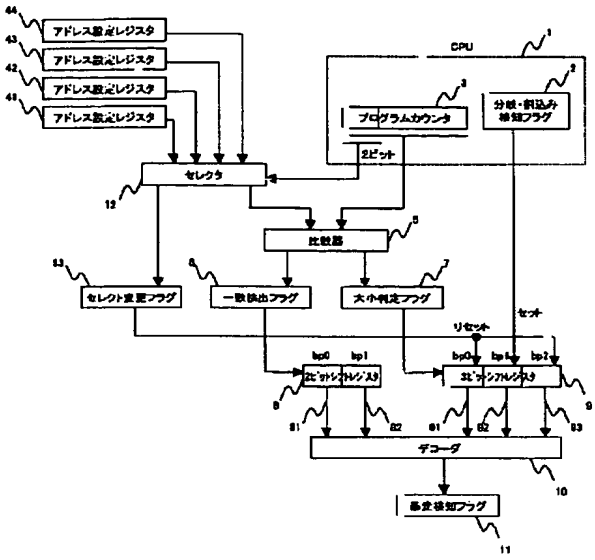
【図４】



【図９】



【図5】



【図6】

分岐の成立や割込みの発生のない状態での正常動作の場合

アドレス設定レジスタ41＝072H
アドレス設定レジスタ42＝114H
アドレス設定レジスタ43＝243H
アドレス設定レジスタ44＝330H

プログラムカウンタ3	セレクタ12	3ビットシフトレジスタ9 (大小判定) bp0 bp1 bp2	2ビットシフトレジスタ8 (一致検出) bp0 bp1
050H	072H	0 0 0	0 0
070H	072H	0 0 0	0 0
072H	072H	1 0 0	1 0
078H	072H	1 1 0	0 1
081H	072H	1 1 1	0 0
セレクト変更		0 1 0	0 0
110H	114H	0 0 1	0 0
114H	114H	1 0 0	1 0
120H	114H	1 1 0	0 1

【図8】

分岐命令が成立した場合

アドレス設定レジスタ41＝072H
アドレス設定レジスタ42＝114H
アドレス設定レジスタ43＝243H
アドレス設定レジスタ44＝330H

プログラムカウンタ3	セレクタ12	3ビットシフトレジスタ9 (大小判定) bp0 bp1 bp2	2ビットシフトレジスタ8 (一致検出) bp0 bp1
050H	072H	0 0 0	0 0
070H	072H	0 0 0	0 0
072H	072H	1 0 0	1 0
078H	072H	1 1 0	0 1
081H	072H	1 1 1	0 0
分岐＋セレクト変更		0 1 0	0 0
250H	243H	1 0 1	0 0
253H	243H	1 1 0	0 0
260H	243H	1 1 1	0 0
分岐＋セレクト変更		0 1 0	0 0
110H	114H	0 0 1	0 0
114H	114H	1 0 0	1 0
120H	114H	1 1 0	0 1

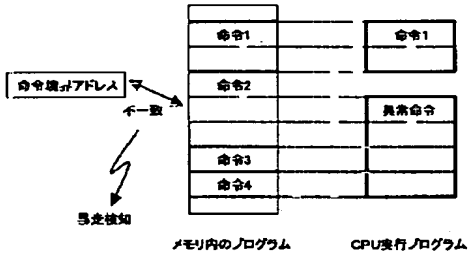
【図7】

異常動作の場合

アドレス設定レジスタ41＝072H
アドレス設定レジスタ42＝114H
アドレス設定レジスタ43＝243H
アドレス設定レジスタ44＝330H

プログラムカウンタ3	セレクタ12	3ビットシフトレジスタ9 (大小判定) bp0 bp1 bp2	2ビットシフトレジスタ8 (一致検出) bp0 bp1	
050H	072H	0 0 0	0 0	
070H	072H	0 0 0	0 0	
072H	072H	1 0 0	1 0	
078H	072H	1 1 0	0 1	
081H	072H	1 1 1	0 0	
セレクト変更		0 1 0	0 0	
110H	114H	0 0 1	0 0	
115H	114H	1 0 0	0 0	暴走検知
120H	114H	1 1 0	0 1	

【図10】



JP 2001-188688

NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]
[Claim 1]A comparator which compares the 1st memory storage for storing data beforehand, and a value of said 1st memory storage with a value of a program counter, and performs a size judgment and coincidence detection, A reckless-run detecting circuit of a microcomputer which consists of a decoder which decodes a value of the 2nd and 3rd memory storage for storing said size judgment and coincidence detecting results, and said 2nd and 3rd memory storage.
[Claim 2]A reckless-run detecting circuit of a microcomputer provided with a means to form two or more 1st memory storage for [said] storing data beforehand.

[Translation done.]

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention]This invention relates to the mechanism for detecting malfunction of a microcomputer.
[0002]
[Description of the Prior Art]Generally, processing follows the microcomputer because CPU executes the instruction code stored in the memory as an order. When the fetch mistake of the abnormalities of a program counter or an instruction code occurs by being influenced by rapid change of a noise or voltage, it becomes impossible however, for CPU to execute a command correctly. In the Prior art, malfunction of CPU has been prevented as much as possible by the surveillance of CPU in a constant interval and the detection of an undefined instruction by a watchdog timer.
[0003]
[Problem(s) to be Solved by the Invention]One of the mechanisms in which a microprocessor with the variable length instruction architecture runs recklessly in response to a noise has the operation to which the value of a program counter becomes unusual. If abnormalities arise in a program counter, data will be taken in from the command boundary mistaken as shown in drawing 9. However, by the time a reckless run is detected as it is a reckless-run detecting mechanism in the constant interval by the conventional watchdog timer, time will be taken, and CPU will malfunction in the meantime. If the instruction code fetched after the reckless run as it is a reckless-run detecting mechanism by detection of an undefined instruction is not against a relation with the instruction code of order, this is undetectable, and the CPU will operate, processing the mistaken command.

[0004]An object of this invention is to detect that the mistaken instruction code was processed in order to solve an aforementioned problem.
[0005]
[Means for Solving the Problem]In order to solve an aforementioned problem, in this invention, as shown in drawing 10, an address used as a command boundary of a variable length instruction is set up beforehand, and it always compares with a program counter. Therefore, an execution gestalt of this invention is equipped with a comparator which compares with said address and a program counter a register which sets up said address. When a program counter becomes an abnormal value to said address, it has a shift register and a decoder for detecting this.
[0006]
[Embodiment of the Invention](Embodiment 1) Drawing 1 is used and explained about a 1st embodiment of this invention below. Drawing 1 is a block diagram showing the composition of the reckless-run detecting circuit in a 1st embodiment of this invention.
[0007]When CPU of a microcomputer generates 1 and a branch instruction or an interrupt instruction generates two, it is branching / interruption detection flag detected within CPU1. 3 is a program counter in CPU1 and 4 is an address selection register for setting up the command boundary address beforehand. The comparator 5 compares the value of the program counter 3 and the address selection register 4.
[0008]From the comparison result of the comparator 5, when the value of the program counter 3 and the value of the address selection register 4 are in agreement, '0' is set to the coincidence detection flag 6 '1' and except it. Similarly, when the value of the program counter 3 is in agreement with the value of the address selection register 4, when large, '0' is set to the size judgment flag 7 '1' and except it.
[0009]8 is two bit shift registers which latch the coincidence detection flag 6, and 9 is three bit shift registers which latch a size judgment flag. By branching / interruption detection flag 2, bp1 equivalent to the 2nd bit of the three bit shift registers 9 is set, when CPU performs a branching process and interrupt processing. The output signals 81 and 82 of the two bit shift registers 8 and the output signals 91, 92, and 93 of the three bit shift registers 9 are decoded by the decoder 10. The decoder 10 generates the reckless-run detection flag 11, when the output signals 91, 92, 93, 81, and 82 become '10000'.
[0010]Below, the flow of reckless-run detection is explained with reference to drawing 2 - drawing 4. Drawing 2 - drawing 4 are the examples which set the command boundary address '53H' (hexadecimal number notation) as the address selection register 4, and, as for drawing 4, drawing 3 shows the case where a branch instruction is materialized [case / of the normal operation in the state where drawing 2 does not have / the operating state of CPU / generating of formation of branching, or interruption] in the case of abnormal operation.
[0011](1) Explain the case where the CPU is operating normally in the case of the normal operation in the state where there is no generating of formation of branching or interruption, with reference to drawing 2. At this time, the value of the program counter 3 changes to the preset value of the address selection register 4 in order of a larger value than a value smaller than a preset value, the same value as a preset value, and a preset value.
[0012]At the time of the value in which the program counter 3 is smaller than the preset value of the address selection register 4, the size judgment flag 7 and the coincidence detection flag 6 are '0', and the three bit shift registers 9 and both the bit shift registers 8 are all the bits '0'. When the value of the program counter 3 is in agreement with preset value'53H' of the address selection register 4, the size judgment flag 7 and the coincidence detection flag 6 become '1', and '1' is latched to bp0 of the three bit shift registers 9 and the two bit shift registers 8. If the value of the program counter 3 turns into a larger value than the preset value of the address selection register 4, the size judgment flag 7 will turn into '1' and the coincidence detection flag 6'0', and as shown in drawing 2, the three bit shift registers 9 will turn into '110' and the

two bit shift registers 8'01'.
[0013]Therefore, in this flow, the output signals 91, 92, 93, 81, and 82 do not become '10000', and the reckless-run detection flag 11 does not become active.
[0014](2) Explain the case where abnormal operation occurs in CPU in the case of abnormal operation, with reference to drawing 3. When the program counter 3 takes an abnormal value by a noise etc., to the preset value of the address selection register 4, the value of the program counter 3 will not be in agreement, and will be passed.
[0015]At the time of the value in which the program counter 3 is smaller than the preset value of the address selection register 4, the size judgment flag 7 and the coincidence detection flag 6 are '0', and the three bit shift registers 9 and both the bit shift registers 8 are all the bits '0'. If it becomes a large value, without the value of the program counter 3 being in agreement with the preset value of the address selection register 4, the size judgment flag 7 will turn into '1' and the coincidence detection flag 6'0', and as shown in drawing 3, the three bit shift registers 9 will turn into '100' and the two bit shift registers 8'00'.
[0016]Therefore, in this flow, since the output signals 91, 92, 93, 81, and 82 become '10000' and the reckless-run detection flag 11 becomes active immediately after the value of the program counter 3 passes the preset value of the address selection register 4, a reckless run by unusual instruction processing is detectable.
[0017](3) When a branch instruction is materialized, explain the case where branching occurs in the operating state of CPU, with reference to drawing 4. The value of the program counter 3 may be passed at this time, without being in agreement with the preset value of the address selection register 4.
[0018]At the time of the value in which the program counter 3 is smaller than the preset value of the address selection register 4, the size judgment flag 7 and the coincidence detection flag 6 are '0', and the three bit shift registers 9 and both the bit shift registers 8 are all the bits '0'. If it becomes a large value, without setting bp1 of the three bit shift registers 9 to '1' by branching / interruption detection flag 2, and the value of the program counter 3 being in agreement with the preset value of the address selection register 4, if branching occurs, The size judgment flag 7 turns into '1' and the coincidence detection flag 6'0', and as shown in drawing 4, the three bit shift registers 9 turn into '101' and the two bit shift registers 8'00'.
[0019]Therefore, in this flow, the output signals 91, 92, 93, 81, and 82 do not become '10000', and the reckless-run detection flag 11 does not become active. Even when an interrupt occurs, it is obvious that same processing is performed.
[0020]Above, from (1), (2), and (3), by this embodiment, if the program counter 3 takes an abnormal value, since the reckless-run detection flag 11 becomes active, reckless-run detection will be attained.
[0021](Embodiment 2) Drawing 5 is used and explained about a 2nd embodiment of this invention below. A 2nd embodiment is a mechanism in which two or more command boundary addresses can be set up. Drawing 5 is a block diagram showing the composition of the reckless-run detecting circuit in a 2nd embodiment of this invention.
[0022]1 is CPU of a microcomputer, and 2 is branching / interruption detection flag detected within CPU1, when a branch instruction or an interrupt instruction occurs. 3 is a program counter in CPU1 and 41-44 are the address selection registers for setting up the command boundary address beforehand. Four address selection registers shall be provided and one command boundary address shall be set up for every memory area distinguished by 2 bits of top of the program counter 3.
[0023]2 bits of top to the address selection register 41 the command boundary address of '00', the address selection register 42 -- 2 bits of top -- the command boundary address of '01' -- the command boundary address of '10' is set to the address selection register 43, and, in 2 bits of top, 2 bits of top set the command boundary address of '11' to the address selection register 44.

[0024]When 2 bits of top of the program counter 3 are '00', the selector 12 the address selection register 41, The command boundary address preset value of the same field as the program counter 3 can be chosen by choosing [the address selection register 42] the address selection register 44 for the address selection register 43 at the time of '11' at the time of '10' at the time of '01'. When selection of the address selection registers 41-44 in the selector 12 changes, '0' is set to the selection change flag 13 '1' and except it.
[0025]The comparator 5 compares the value of the program counter 3 and the selector 12. From the output of the comparator 5, when the value of the program counter 3 and the value of the selector 12 are in agreement, '0' is set to the coincidence detection flag 6 '1' and except it. Similarly, when the value of the program counter 3 is in agreement with the value of the selector 12, when large, '0' is set to the size judgment flag 7 '1' and except it. 8 is two bit shift registers which latch the coincidence detection flag 6, and 9 is three bit shift registers which latch a size judgment flag. By branching / interruption detection flag 2, bp1 of the three bit shift registers 9 is set, when CPU performs a branching process and interrupt processing, and bp0 and bp2 are cleared by the selection change flag 13. The output signals 81 and 82 of the two bit shift registers 8 and the output signals 91, 92, and 93 of the three bit shift registers 9 are decoded by the decoder 10. The decoder 10 generates the reckless-run detection flag 11, when the output signals 91, 92, 93, 81, and 82 become '10000'.
[0026]Since a command boundary address can be set up for every memory area distinguished at top n bits of the program counter 3, n address selection 2 registers can also be provided.
[0027]Below, the flow of reckless-run detection is explained with reference to drawing 6 - drawing 8. Drawing 6 - drawing 8 to the address selection register 41 as a command boundary address '072H', To the address selection register 42 at '114H' and the address selection register 43 '243H', It is the example which set '330H' as the address selection register 44, and, as for drawing 8, drawing 7 shows the case where a branch instruction is materialized [case / of the normal operation in the state where drawing 6 does not have / the operating state of CPU / generating of formation of branching, or interruption] in the case of abnormal operation.
[0028](1) Explain the case where the CPU is operating normally in the case of the normal operation in the state where there is no generating of formation of branching or interruption, with reference to drawing 6. At this time, the value of the program counter 3 changes to the selection value of the selector 12 in order of a larger value than a value smaller than a selection value, a selection value, and a selection value.
[0029]By drawing 6, the program counter 3 shows the example which begins from '050H'. Since the address selection register 41 is chosen by 2 bits of top '00' of the program counter 3 by the selector 12 at this time, the selection value of the selector 12 is '072H'. At the time of the value in which the program counter 3 is smaller than the selection value of the selector 12, the size judgment flag 7 and the coincidence detection flag 6 are '0', and the three bit shift registers 9 and both the bit shift registers 8 are all the bits '0'. When the value of the program counter 3 is in agreement with selection value'072H' of the selector 12, the size judgment flag 7 and the coincidence detection flag 6 become '1', and '1' is latched to bp0 of the three bit shift registers 9 and the two bit shift registers 8.
[0030]If the value of the program counter 3 turns into a larger value than the selection value of the selector 12, the size judgment flag 7 will turn into '1' and the coincidence detection flag 6'0', and as shown in drawing 6, the three bit shift registers 9 will turn into '110' and the two bit shift registers 8'01'.
[0031]By the selection change flag 13 being set to '1', if a program furthermore progresses and 2 bits of top of the program counter 3 change from '00' to '01'. bp0 and bp2 of the three bit shift registers 9 are cleared by '0', and selection of the selector 12 changes to the address selection register 42.
[0032]Therefore, in this flow, the output signals 91, 92, 93, 81, and 82 do not become '10000', and the reckless-run detection flag 11 does not become active.

[0033](2) Explain the case where abnormal operation occurs in CPU in the case of abnormal operation, with reference to drawing 7. When the program counter 3 takes an abnormal value by a noise etc., to the selection value of the selector 12, the value of the program counter 3 will not be in agreement, and will be passed.
[0034]The program counter 3 presupposes that it progressed to '110H' like the case of the normal operation of drawing 6. Then, when it becomes a large value, without the program counter's 3 taking an abnormal value and being in agreement with the selection value '114H' of the selector 12, as the size judgment flag 7 turns into '1' and the coincidence detection flag 6'0' and it is shown in drawing 7, The three bit shift registers 9 turn into '100' and the two bit shift registers 8'00'.
[0035]Therefore, in this flow, immediately after the value of the program counter 3 passes the selection value of the selector 12, the output signals 91, 92, 93, 81, and 82 become '10000', the reckless-run detection flag 11 becomes active, and a reckless run by unusual instruction processing can be detected.
[0036](3) When a branch instruction is materialized, explain the case where branching occurs in the operating state of CPU, with reference to drawing 8. The value of the program counter 3 may be passed at this time, without being in agreement with the selection value of the selector 12.
[0037]The program counter 3 presupposes that it progressed to '081H' like the case of the normal operation of drawing 6. By then, the thing for which the selection change flag 13 will be set to '1' if branching occurs, and 2 bits of top of the program counter 3 change to '10' from '00'. bp0 and bp2 of the three bit shift registers 9 are cleared by '0', and bp1 is further set to '1' by branching / interruption detection flag 2. Selection of the selector 12 changes to the address selection register 43. When it becomes a large value by this branching, without the value of the program counter 3 being in agreement with the selection value of the selector 12, as the size judgment flag 7 turns into '1' and the coincidence detection flag 6'0' and it is shown in drawing 8, the three bit shift registers 9 -- '101' and the two bit shift registers 8'00 -- ' -- it becomes.
[0038]Therefore, in this flow, the output signals 91, 92, 93, 81, and 82 do not become '10000', and the reckless-run detection flag 11 does not become active. Even when an interrupt occurs, it is obvious that same processing is performed.
[0039]Accurate reckless-run detection is attained by providing two or more address selection registers from (1), (2), and (3) above at this embodiment.
[0040]
[Effect of the Invention]Since reckless-run detection can be performed according to this invention when a program counter takes an abnormal value by the noise, a rapid voltage variation, etc. so that an above embodiment may show, malfunction of CPU can be prevented.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1]The block diagram showing the composition of the reckless-run detecting circuit in a 1st embodiment of this invention
[Drawing 2]The figure showing an example (in the case of the normal operation in the state where there is no generating of formation of branching or interruption) of the operation flow of the reckless-run detecting circuit in a 1st embodiment of this invention
[Drawing 3]The figure showing an example (in the case of abnormal operation) of the operation flow of the reckless-run detecting circuit in a 1st embodiment of this invention
[Drawing 4]The figure showing an example of the operation flow of the reckless-run detecting circuit in a 1st embodiment of this invention (when a branch instruction is

materialized)
[Drawing 5]The block diagram showing the composition of the reckless-run detecting circuit in a 2nd embodiment of this invention
[Drawing 6]The figure showing an example (in the case of the normal operation in the state where there is no generating of formation of branching or interruption) of the operation flow of the reckless-run detecting circuit in a 2nd embodiment of this invention
[Drawing 7]The figure showing an example (in the case of abnormal operation) of the operation flow of the reckless-run detecting circuit in a 2nd embodiment of this invention
[Drawing 8]The figure showing an example of the operation flow of the reckless-run detecting circuit in a 2nd embodiment of this invention (when a branch instruction is materialized)
[Drawing 9]The figure showing the example of unusual instruction processing depended unusually [a program counter]
[Drawing 10]The key map of reckless-run detection twisted unusually [the program counter of this invention]
[Description of Notations]
1 CPU
2 Branching / interruption detection flag
3 Program counter
4 Address selection register
5 Comparator
6 Coincidence detection flag
7 Size judgment flag
8 Two bit shift registers
9 Three bit shift registers
10 Decoder
11 Reckless-run detection flag
12 Selector
13 Selection change flag
41 Address selection register
42 Address selection register
43 Address selection register
44 Address selection register
81 2 bit-shift-register 8 output signal
82 2 bit-shift-register 8 output signal